**În C++ variabilele pot fi împărţite în 3 mari categorii: locale, globale şi dinamice**

Sistemul de operare alocă fiecărui program **trei zone distincte în memoria internă** în care se găsesc memorate variabilele programului:

1. **Segment de date ( variabile globale sau cele declarate static/**pag139manual**)**
2. **Segment de stivă ( variabile locale/ subprograme)**
3. **Heap. ( variabile alocate dinamic de utilizator )**

Există şi posibilitatea ca variabilele să fie memorate într-un anumit **registru** al microprocesorului, caz în care accesul la acestea este foarte rapid. *(vezi pag 140 manual)*

**O variabilă se caracterizează prin 4 atribute:**

o **Clasa de memorare** – locul unde este memorată variabila respectivă; o variabilă poate fi memorată în:

* segmentul de date (variabilele globale)
* segmentul de stivă (în mod implicit, variabilele locale)
* heap
* un registru al microprocesorului (în mod explicit, variabilele locale).

o **Vizibilitatea** – precizează liniile textului sursă din care variabila respectivă poate fi accesată. Există următoarele tipuri de vizibilitate:

* la nivel de bloc (instrucţiune compusă) (variabilele locale);
* la nivel de fişier sursă (în cazul în care programul ocupă un singur fişier sursă) (variabilele globale, dacă sunt declarate înaintea tuturor funcţiilor);
* la nivel de clasă (în legătură cu programarea orientată pe obiecte).

o **Durata de viaţă** – timpul în care variabila respectivă are alocat spaţiu în memoria internă. Avem:

* durată **statică** – variabila are alocat spaţiu în tot timpul execuţiei programului (variabilele globale);
* durată **locală** – variabila are alocat spaţiu în timpul în care se execută instrucţiunile blocului respectiv (variabilele locale);
* durată **dinamică** – alocarea şi dezalocarea spaţiului necesar variabilei respective se face de către programator prin operatori specifici;

o **Tipul variabilei**

|  |  |  |  |
| --- | --- | --- | --- |
| **pointeri**  #include <iostream.h>  void schimba**(int \*a, int \*b)**  { int aux = \*a;  \*a = \*b;  \*b = aux;  }  main() {  int x = 2, y = 3;  schimba**(&x, &y**);  cout<< x << " " << y;  } | ***Analizati efectul celor 3 subprograme- (funcții procedurale)*** | | |
| #include <iostream.h>  void schimba**(int a, int b)**  { int aux = a;  a = b;  b = aux;  }  main() {  int x = 2, y = 3;  **schimba(x, y**);  cout<< x << " " << y;  } | | #include <iostream.h>  void schimba**(int &a, int &b)**  { int aux = a;  a = b;  b = aux;  }  main() {  int x = 2, y = 3;  **schimba(x, y);**  cout<< x << " " << y;  } |
| **int** prim(int n)  { int i;  for (i = 2; i<= n/2; i++)  if(n%i==0) return 0;  return 1; } | | Apel posibil- **funcții operand**   1. **Atribuire** x=prim(n); 2. **Decizie**   If(prim(vb))….   1. **Afisare**   Cout<<prim(8); | |
| Observații-**Parametrii actuali**( efectivi/ la apel):   1. corespunzători param.transmiși prin valoare pot fi exprimați prin:valoare, expresie, variabila de memorie(identificator data) sau adresa unei vb. de memorie la cei de tip pointeri 2. la cei corespunzători param.transmiși prin referință pot fi exprimați numai prin variabile de memorie | | | |